## REMARKS

Independent claims 1, 8 and 16 have been amended for clarity, to insert a limitation previously included in claim 20. Claim 8 has been amended so that the apparatus thereof is infringed without a set of target instructions being included therein. Claims 12 and 13 have been amended for clarity and consistency. Claim 15 has been amended as suggested by the examiner. Claims 21-23 have been added to provide applicant with the protection to which he is deemed entitled. The operations defined by claims 21-23 are disclosed on pages 11 and 12 of the application as filed.

Applicant traverses the rejection of claims 8-14 and 16- 20 under 35 USC 101. The office action alleges claims 8-14 and 20 are directed to software per se, not tangibly embodied and claims 16-19 are directed to software per se, claiming only coded indicia. Claims 8-14 are directed to an apparatus, that is, a machine, for generating code for scheduling the execution of binary code translated from a source format to a target format, while claim 20 is concerned with an apparatus for translating machine instructions in source code into equivalent target instructions of a code of a target platform. An apparatus is not software. The instruction analyzer and the dependency identifier of independent claim 8, upon which claims 9-14 and 19 depend, are not necessarily software. In addition, the templates, fill and analysis routine generator and dynamic binary translator of claim 20 are not necessarily software. Claim 13 is not directed to software because it concerns a directed acyclic graph. Claim 16, upon which claim 17 depends, is directed to an article of manufacture because it defines a computer-readable medium or storing device. Claim 18, which depends on claim 1, concerns operation of the code analyzer of claim 1. Based on the foregoing, claims 8-14 and 16-20 conform with 35 USC 101.

Applicant traverses the rejection of claims 1-4, 7-11 and 14-19 under 35 USC 103(a) as being unpatentable over Hughes et al. US Patent 6,519,768 in view of Bharadwaj US Patent 5,894,576. Claim 1 is concerned with a method of generating code for scheduling the execution of binary code translated from a source format to a target format, while independent claim 8 is directed to apparatus for generating code for

scheduling the execution of binary code translated from a source format to a target format. There is no consideration in the office action to these requirements of claims 1 and 8. Applicant is unable to find anything dealing with scheduling the execution of binary code translated from a source format to a target format in either reference. While Hughes et al. is concerned with translating source code instructions into target code instructions and Bharadwaj relates to scheduling instructions, neither reference deals with the scheduling problem associated with claims 1 and 8.

The office action erroneously alleges Hughes et al., at column 3, lines 57-65, meets the requirements of claims 1 and 8 to assign an identifier to one or more target instructions for use by a code analyzer in scheduling the processing of a set of target instructions. There is no mention of scheduling in this portion of Hughes et al.. Instead, this portion of Hughes et al. indicates, that for each marker value in a template, the Initialize Templates process inserts a fix-up entry into data structures. The fix-up entry identifies the location of a marker value and specifies the data type of a constant value that is to be inserted into a template source code at translate time, that is, run time, to replace the marker value. For each call in the template, the initialize templates process inserts a fix-up entry in the data structures, thereby identifying the location of the call. It is not seen how this has anything to do with assigning an identifier in connection with scheduling the processing of a set of target instructions.

The office action admits Hughes et al. fails to disclose the requirements of claims 1 and 8 to identify data dependencies in target instructions by analyzing the set of target instructions. The office action fails to consider the requirement of claims 1 and 8 for the scheduling of the processing of the set of target instructions to be in accordance with the identified data dependencies. Because Hughes does not analyze a set of target instructions, as admitted in the office action, Hughes et al. cannot make obvious the scheduling of the processing of the set of target instructions in accordance with the identified data dependencies.

The office action also erroneously alleges Hughes et al., at column 2, lines 5-8, discloses the requirements of claims 1 and 8 to identify a set of target instructions semantically equivalent to a given source instruction. In this regard, the office action mis-paraphrases column 2, lines 5-8 by alleging this portion of Hughes et al. discloses "utilizing a template and applying it to block of target instructions corresponding to a particular source code instruction. (sic)" In fact, column 2, lines 5-8 of Hughes et al. states: "(b) for each instruction in an input block of source code instructions, selecting an appropriate template for that source code instruction and appending this template to an output block of target code instructions;". In the opinion of applicant, the paraphrasing in the office action does not correspond with what is stated in column 2, lines 5-8 of Hughes et al.. The examiner is requested to explain how his paraphrasing of column 2, lines 5-8 corresponds with what is actually stated in this portion of the reference.

The office action alleges Bharadwaj is in an analogous art to Hughes et al.. Applicant cannot agree because Bharadwaj has nothing to do with the thrust of the Hughes et al. reference, which deals with translating instructions, and more particularly to translating a source code instruction into a target code instruction. Column 1, lines 7-12 of Bharadwaj indicates the reference is concerned with a method of instruction scheduling to enable the harmful effects of compensation code to be reduced. Such technology is quite far afield from the Hughes et al. code translator.

The office action alleges Bharadwaj, at column 6, lines 57-64, discloses identifying data dependencies in a target instruction by analyzing a set of target instructions and assigning identifiers in accordance with identified data dependencies. However, the relied on proportion of Bharadwaj states that data dependency analyzer 68 generates a directed acyclic graph representation of instructions in a region by using control flow data. The directed acyclic graph gives a hierarchy of data dependence between instructions, showing how the instructions depend on the outcome of previous instructions. The examiner is requested to indicate how this portion of Bharadwaj meets the requirements of claims 1 and 8 to identify data dependencies in target instructions by analyzing a set of target instructions.

The office action alleges it would have been obvious to have modified Hughes et al. as a result of Bharadwaj "to increase the overall efficiency of translating code, as one would be motivated to make combination for the benefit of reducing the number of execution cycles needed as a result." Since Bharadwaj is not concerned with translating code, one of ordinary skill in the art would not have modified Hughes et al. as a result of Bharadwaj. In addition, the examiner must explain how one of ordinary skill in the art would know from Hughes et al. and Bharadwaj that there would be a reduction in the number of execution cycles. Hence, the rejection of claims 1 and 8 based on Hughes et al. and Bharadwaj is erroneous.

Claims 2-4, 7, 15 and 18, which depend on claim 1, and claims 9-11, 14 and 19, which depend on claim 8, all of which are rejected on Hughes et al. in combination with Bharadwaj, are patentable for the same reasons advanced supra with regard to claims 1 and 8.

In addition, the allegation in the office action that Bharadwaj, at column 4, line 61-column 5, line 14, and column 6, lines 57-64, discloses the requirement of claims 18 and 19 for a code analyzer to schedule the processing of a set of target instructions in accordance with identified data dependencies is erroneous. Column 6, lines 57-64 has been previously discussed and indicated to be irrelevant to the foregoing requirements of claims 18 and 19. Column 4, line 61-column 5, line 14 of Bharadwaj indicates Figure 6 of the reference is a block diagram of one embodiment of the scheduler of Figure 5. The scheduler is indicated as including a region builder, a control flow analyzer, a data dependence analyzer, an instruction prioritizer, a data readiness analyzer, a wavefront initializer, an instruction scheduler and a wavefront updater. At the beginning of a scheduling process, the region builder partitions the program into regions. A region is the portion of a control flow diagram over which scheduling is performed and can have a number of entry and exit points. Control flow within a region must be acyclic so the region cannot directly contain loops. The region may contain special blocks representing nested loops or nested regions, with loop nesting represented through a hierarchy in the regions. Regions formed for acyclic control flow can also be organized in a hierarchical structure to

force an order in the scheduling, to allow scheduling across inner regions and to enable reasons to be formed with outside entries. The examiner is requested to indicate how the foregoing meets the requirements of claims 18 and 19 for a code analyzer to schedule the processing of a set of target instructions in accordance with identified data dependencies.

The rejection of claim 16 relies on column 2, lines 5-8 of Hughes et al. to disclose utilizing a template and applying it to a block of target instructions corresponding to a protector source code instruction to identify semantically equivalent target and source instructions. Applicant has previously indicated, in connection with the rejection of claims 1 and 8, why the examiner has misinterpreted this portion of Hughes et al.

The rejection of claim 16 admits Hughes et al. fails to disclose the requirement of claim 16 for a set of analysis routines arranged to identifie data dependencies in a template for causing generation of data for use by a code scheduler in scheduling the execution of translated code on a target processor. The office action relies on Bharadwaj for this feature. However, as discussed supra, one of ordinary skill in the art would not have relied on Bharadwaj to modify Hughes et al. because they are not concerned with the same technologies, except from a very broad standpoint. In addition, Bharadwaj does not identify data dependencies in a template for causing generation of data for use by a code scheduler in scheduling the execution of translated code on a target processor. The office action relies on column 6, lines 57-64 of Bharadwaj for this feature. However, the previous discussion of column 6, lines 57-64 indicates it is not germane to identifying data dependencies as set forth in claim 16. Based on the foregoing, the rejection of claim 16 is wrong.

Applicant traverses the rejections of claims 5 and 12, respectively dependent on claims 1 and 8, based on Hughes et al., Bharadwaj, and Mochizuki, US Patent 6,016,396 and of claims 6 and 13, respectively dependent on claims 1 and 8, based on Hughes et al., Bharadwaj, and Rasbold, US Patent 5,202,975. The tertiary references fail to cure the above noted deficiencies in the rejections of the independent claims. Consequently, claims 5, 6, 12 and 13 are allowable.

Applicant traverses the rejection of claim 20 under 35 USC 103(a) as being unpatentable over Hughes et al. In this rejection, the office action admits Hughes et al. fails to disclose a fill and analysis routine generator arranged to be responsive to the templates and generating fill and analysis routines for identifying fillable positions in a template by parsing the template after generating code to extract and deposit fields from machine instructions in source code into a precompiled template. The office action also admits Hughes et al. fails to disclose parsing the template.

The office action states that analyzing templates for adding or removing functions is analogous to a fill and analysis routine used in template analyzing. The office action, however, provides no rationale as to why the Hughes system for analyzing templates for adding or removing functions is analogous to the claimed fill and analysis routines. Consequently, the office action incorrectly rejects claim 20 as being obvious as a result of Hughes et al. The office action also alleges parsing is well-known knowledge of the art, but fails to indicate that parsing is well-known in the specific art relating to dynamically translating a source code instruction into equivalent target code instructions, as required by claim 20. Consequently, one of ordinary skill in the art would not have used parsing to break down a code. It remained for applicant to realize the advantages of the features of claim 20 that the examiner admits Hughes et al. fails to disclose.

In view of the foregoing amendments and remarks, allowance is in order.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due in connection with the filing of this paper, including extension of time fees, to Deposit Account 08-2025 and please credit any excess fees to such deposit account.

Respectfully submitted,

**LOWE HAUPTMAN HAM & BERNER, LLP**

Dibyapran SANYAL

/Allan M. Lowe/

Allan M. Lowe
Registration No. 19,641

**HEWLETT-PACKARD COMPANY**
Intellectual Property Administration
P. O. Box 272400
Fort Collins, CO 80527-2400
703-684-1111 Telephone
970-898-0640 Telecopier
**Date: June 5, 2008**
AML/cjf